(12)

RADC-TR-82-319
In-House Report
December 1982

USING SREM TO SPECIFY COMMAND AND CONTROL SOFTWARE REQUIREMENTS

William E. Rzepka

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441

RADC-TR-82-319 has been reviewed and is approved for publication.

APPROVED: [illegible]
DEANE F. BERGSTROM
Chief, Computer & Software Engineering Branch
Command and Control Division

APPROVED: [illegible]
JOHN J. MARCINIAK, Colonel, USAF
Chief, Command and Control Division

FOR THE COMMANDER: John P. Huss
JOHN P. HUSS
Acting Chief, Plans Office

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-82-319	2. GOVT ACCESSION NO. A124429	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) USING SREM TO SPECIFY COMMAND AND CONTROL SOFTWARE REQUIREMENTS		5. TYPE OF REPORT & PERIOD COVERED In-House Report 1 Oct 79 - 30 Sep 82
		6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) William E. Rzepka		8. CONTRACT OR GRANT NUMBER(s) N/A
9. PERFORMING ORGANIZATION NAME AND ADDRESS Rome Air Development Center (COEE) Griffiss AFB NY 13441		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS J.O. 55812205
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (COEE) Griffiss AFB NY 13441		12. REPORT DATE December 1982
		13. NUMBER OF PAGES 21
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Same		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

SREM, software requirements, specification language, command and control system.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report briefly describes command and control systems and the documentation standards currently used to specify their embedded computer software requirements. The Software Requirements Engineering Methodology (SREM) appears to satisfy these documentation standards. However, the original SREM design did not specifically address command and control needs, thereby leaving uncertain the exact extent of the SREM's capabilities to specify problems unique to this domain. To resolve this

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

issue, the SREM is used to specify the software requirements for a small, but representative, command and control problem. The results show that, in general, SREM is capable of specifying command and control software requirements, but that certain problems exist. Some of the problems are easily resolved using the self-extension mechanisms of SREM, but other problems are sufficiently serious to warrant further investigations involving full-scale command and control systems.

TABLE OF CONTENTS

SECTION 1

INTRODUCTION

A command and control (C2) system is a collection of people, procedures and equipment organized to perform a specific mission. The mission may involve activities such as target tracking and identification, surveillance data acquisition, communications processing, network control, mission planning, operations monitoring, and the equipment can involve aircraft, sensors, communication devices and computers [CLAP77]. The computers play a vital role in command and control systems because they provide the commander with the capability to manage a large and complex system and enable him to respond to dynamic situations with informed decisions. These capabilities are possible because of the power and flexibility of the computer software, but not without the significant costs of developing these large and complex programs. Managing software developments requires a disciplined and structured approach, beginning with a definition of software requirements - what the software must do in order to accomplish the mission.

This paper will describe the software requirements specification

needs of command and control systems, their requirements specification problems and how the problems can be alleviated. The Software Requirements Engineering Methodology (SREM) is discussed as an approach for specifying command and control software requirements. The results of a study of its responsiveness to specifying command and control software requirements are presented. These results are the principal inputs to an Air Force sponsored program for further intensive evaluation of SREM in the command and control context. This program is described and its current status is reported.

SECTION 2

C2 SOFTWARE REQUIREMENT SPECIFICATION NEEDS

Regulations [USAF75] govern the process of acquiring military command and control systems. These regulations include standards for developing and documenting the various system components. For computer software the regulations prescribe a development process with formal phases which are delineated by milestones. At each milestone reviews are held and documentation and/or software are delivered.

The process begins with software requirements analysis. This activity determines what functions the software will perform. The inputs to this activity are the system functions which have been assigned to the computer software during system requirements analysis. The results of the software requirements analysis are documented and input to the next development phase - software design. The design activity differs from its predecessor in that it determines how the software will be organized and written. Following design, software modules are coded, unit tested and then integrated into a complete system.

Software requirements analysis is a critical step, because it sets the course for all software development to follow. For military command and control systems software requirements are documented in accordance with standards, such as MIL-STD 490 [MILS68]. Although several standards exist, in general, they require that:

the software's functions must be described;
the software's performance capabilities must be specified;
the tests which will demonstrate acceptable function and performance must be defined.

Documentation requirements for command and control systems are usually satisfied with a specification which is primarily an English language description of the software requirements supplemented with appropriate graphics. Because these English language specifications describe large and complex systems, they are frequently ambiguous, inconsistent and untestable. It is also difficult to maintain their relationships to both originating (system) requirements and to corresponding software design elements [BELL76]. The software produced from these faulty specifications exhibits many errors during integration of the programs and in the delivered system. As a result, the cost of software development and maintenance is significantly increased. One study [BOEH76] estimates the cost of fixing an error in fielded software as 20 times greater than if the error had been detected and corrected as a requirements problem.

One proposed solution to these problems is the Software Requirements Engineering Methodology. This approach offers a set of guidelines for organizing software requirements into a logical structure called a Requirements Network (R-Net). A Requirements Specification Language is used to formally specify the structure of the R-Nets and the data which they process. A processor for the language translates the specification into a relational database. The stored relationships can be analyzed for consistency and completeness with a database query tool called the Requirement Engineering Validation System (REVS).

SREM encourages the organization of software requirements into a sequence of functions which are the computer's response to stimuli received from external subsystems. This stimulus-response or data flow approach is attractive for command and control systems because it satisfies all three requirements (function, performance and testing) of the documentation standards currently used. The stimulus-response approach adequately describes software functions as a sequence of processing steps - a path - which transforms data inputs into required outputs. Stimulus-response descriptions are also ideally suited to performance specification, because the path or a part of it (sub-path) can be readily identified with resource (cpu, memory) utilization. The paths and sub-paths are defined by end points which identify data which can be utilized to determine if the software meets the performance requirements.

The SREM's ability to satisfy all three elements of the documentation has been described in two studies ([STAI76], [MCDO78]). Furthermore, these studies pointed out the advantages of SREM's detailed work procedures which guide the development of software requirements, the flexibility of its language which can be readily extended to accommodate new specification needs and the generality of its analysis tools which provide a full Boolean query capability on the requirements data base.

Despite the fact that the studies indicated that SREM could satisfy existing software requirements specification standards and that there was some experience [SLEG] in using SREM on a command and control application, other factors left uncertain the exact extent of SREM's capabilities to specify problems unique to this domain. First, SREM was originally developed for application to ballistic missile defense systems which are characterized by a single computer environment with stringent real-time processing requirements. Although command and control systems exhibit real-time requirements there is an emphasis on multiple computers, extensive man-machine interactions and large databases. Second, there is also a need for this kind of technology to assist in its independent verification and validation of command and control software requirements. Third, there was the need to address several "lessons learned" from recent experiences [GREW80] with attempting to transfer similar technology into the command and control system acquisition process. These experiences indicated that the user interface, training and costs of requirements analysis tools are on the critical path to their successful usage on large software acquisition projects. The user interface must be "friendly", with emphasis on graphic representation. It must provide aids (templates, prompting) to assist in preparation of complex requirements language statements and its output reporting tools must be sufficiently flexible to respond to new database queries without requiring re-programming. Well organized and documented user training is mandatory because of the complexity of specification methodologies and their tools. Because command and control systems are large and complex, their software requirement databases are also sizeable. As a result, the requirements analysis tools which manipulate the database must do so efficiently, or the costs of using them will quickly become excessive.

These considerations indicated a need for more information about SREM in the command and control context.

SECTION 3

IN-HOUSE SREM EVALUATION

The purpose of this study was to determine if SREM is a viable approach for specifying command and control software requirements; that is, does it have sufficient specification mechanisms. Identification of command and control problems to which SREM is unresponsive or only partially responsive was another objective. The evaluation project also served as a means for familiarizing project personnel with the SREM, its training, documentation and tools.

The evaluation approach selected was that of validating an existing set of software requirements. Besides allowing full exposure to the SREM and its tools, this approach provided experience with SREM in the independent verification and validation role. Also, a validation procedure does not require the kind of expertise in the application area as does the procedure of generating requirements. As a result, more time could be devoted to the actual evaluation of SREM.

The application selected was a computer based system of tools for simulating the movements, communications and deployment behavior of military units within a particular geographic area and under varying atmospheric conditions. It is called Computer Aided Scenario Development (CASD) and is part of a simulation system for assessing the effectiveness of sensor configurations.

The in-house study consisted of attending the SREM training course and then applying SREM to the validation of the CASD software requirements. The requirements were identified from an existing CASD specification, graphically described using R-Networks, translated into the RSL and analyzed using the REVS. The results fell into two categories: general observations on the effectiveness of SREM and specific problems resulting from the CASD application.

3.1 General Observations

In general, SREM was able to satisfy all of the specification requirements of this kind of command and control problem. Its R-Networks and RSL adequately described the main CASD components:

man-machine interface - commands for developing, maintaining and executing a simulation;
database - files containing the data structures and programs which comprise a simulation;
external communications - messages from external subsystems which provide input data (military doctrine, environment factors) to CASD and messages to external subsystems (e.g.,sensor simulators) which utilize the military scenarios generated by the CASD.

The RSL also provided convenient traceability mechanisms in the CASD database for associating software requirements and their originating system requirements.

Some problems were noted in the areas of terminology, documentation and examples. SREM uses the term "entity" to denote a class of real world objects about which the computer must record and maintain data. The term "entity-type" is used to denote one of the several states which an object can be in, depending upon the particular data associated with the object. The term "type", however, is commonly used to refer to the design concept of a data object and the allowable operations on that object. Although the entity terminology is well-defined and self-consistent within SREM, it conflicts with common usage of the terms, distracts the student during the learning process and makes difficult the explanation of a basic SREM idea. Adding to the confusion is an RSL syntax which appears to specify operations, such as create and delete, on all of the members of an entity class, when the intent is only to affect a specific instance within the class.

In general, SREM documentation is voluminous, especially the training course materials. To err on the side of verbosity is certainly preferable in this case. Unfortunately, indexing of the training materials is an "exercise left to the student."

Current documentation contains several detailed examples of the application of SREM to small systems. However, none of these are command and control systems. During the CASD evaluation, examples identifying basic command and control entity-classes (e.g., files) and their entity-types (e.g., in use, not in use) would have been very helpful. Examples of SREM applied to command and control problems are needed, since they serve as models for applying SREM to new classes of problems and are an important learning mechanism.

It was also noted that the degree to which a SREM description of software requirements succeeds in communicating what needs to be done to the software designer is questionable. There are several reasons for this. Certainly the size and complexity of command and control systems is a factor, as is the translation which the designer must make from an RSL data flow description of requirements to a structured software design characterized by hierarchical organization, abstraction, modularity and information hiding. Complicating these matters is the inability of SREM's data flow approach to provide the designer with a representation of the overall organization of the software requirements - an equivalent of the top level module of a design done using the iterative, step-wise refinement technique.

3.2 CASD Application Results

During the application of SREM to the CASD, problems were encountered in the specification of the man-machine interface, database and communications areas. The man-machine interface consisted of an analyst at a keyboard/display terminal. In creating simulations the analyst will typically perform an operation on files, the exact number of which often depends on the kind of scenario being created. This generates requirements to specify input messages with a variable number of arguments and data elements which are names of objects, in this case, names of files. Because the SREM makes the implicit assumption that the messages transmitted between an external device (e.g., keyboard/display terminal) and the computer are the actual data to be processed, its specification language, RSL, was not originally intended to represent names of objects as objects themselves. Under that assumption there was also no need to express a message which might have a different number o data components on each instance of its use.

The design of RSL is, however, sufficiently flexible to accommoda these CASD requirements. Because RSL is an extendible language, a new data type, character, was introduced to represent file names. Messages with a variable number of data components were (somewhat awkwardly) represented by a file, each of whose records contained one of the components, in this case, the filename arguments.

The *database* to be specified in the CASD application consisted of several files containing the data and source programs which are compiled and link edited by the analyst to form a scenario. The data and programs in the individual files are not logically connected to each other until they are brought together in the new file to form the simulation. As a result, the main CASD database specification requirement is to represent the data contained in individual files. This is easily done in RSL, as long as the data can be represented in simple, record format. Some CASD files, however, required the representation of repeating groups - records whose component elements may contain one or more occurrences of that element. For example, the file which contained all of the programs that made up a scenario, contained a record for each program. Within each record was stored the name of the program plus several other repeating groups of information, such as the program's parameters and its data types. Clearly, the number of parameters or types will vary from program to program, so the repeating group structure provides a straightforward means of representing this requirement. Unfortunately, RSL has no direct way of expressing this structure. In the CASD application it was represented by a file whose records contained data elements which were the names of other files which actually contained the repeating group information.

There are two kinds of communication requirements in the CASD application: communications to external subsystems (e.g., analyst console, sensor simulator) and internal communications among CASD processes. External communications are conveniently represented using the RSL "subsystem" and "message" constructs; that is, the particular device is specified as a "subsystem" and data communications between the device and the computer are specified as "messages."

Although not a CASD requirement, communication between external subsystems is a general requirement which would be encountered in the specification of requirements for a command and control system, such as the sensor configuration assessment system of which the CASD is a part. RSL does not contain mechanisms for direct communications between subsystem elements, because it was designed to specify systems whose components communicate through a single centralized computer. The sensor configuration assessment system consists of several computers. Software requirements for the subsystems resident on each of the computers could be represented in RSL in accordance with the SREM. However, this causes name duplication problems at the computer interfaces. For example, scenario data is generated by the CASD and is then sent to the sensor simulator. It would be specified as both an output (interface) message with respect to the CASD and as an input (interface) message with respect to the sensor simulator. But in each case it is the same data with the same name. Making the names unique results in an equally confusing and undesirable specification which refers to the same message by two different names. If the software requirements for each computer were maintained in a separate database, then the name duplication problem would be avoided. However, the benefits of a single repository for all of the software requirements, making possible consistency checking and traceability analyses, are lost. Further investigation of an approach for the use of the SREM on multiple computer systems is needed.

The internal communications requirements of the CASD are generated by two processes which run the scenario simulation and provide the analyst with on-line control over its execution. These processes need to signal to each other the occurrence of events and to wait for the occurrence of those events. The SREM and RSL provide mechanisms for R-Networks to trigger the occurrence of events and to enable other R-Networks to respond to them. However, the SREM was never intended to deal with the general problem of interprocess communication, because the problem is usually thought of as a design issue. In the CASD context, what is required is a sequence of processing steps which will transform analyst commands into the execution and control of a simulation which generates scenario data usable by other subsystems. How this is done is through the actions of several processes whose cooperation is assured through the mechanism of interprocess communication.

The occurrence of the CASD interprocess communication "requirement" is but one example of a design issue appearing in a software requirements specification. Message formats and physical database organizations are also commonly encountered. There are several directions that a SREM validation methodology may take to resolve this anomaly. First, the design issues could be ignored during the requirements analysis with the intention that their logic be studied during design validation. Second, the validation approach could be extended to accommodate design issues as they arise. Since there is no guarantee that the same design issues would appear in every specification, this choice would result in either a very large, requirements-design language or in many application specific variations of a requirements specification language. Past language experience indicates that neither result is particularly appealing. Third, the software engineers performing the validation could formulate a set of requirements to correspond to the design. For example, only logical database organizations would be represented. The problem here is to generate requirements which would satisfy the intent of the specification writer who proposed the original design. From this range of alternatives and their implications, it is clear that a SREM validation methodology requires further, careful thought.

3.3 SREM Evaluation Conclusions

Applying SREM to the CASD problem provided valuable experience and insight into the use of the methodology, specification language and analysis tools. Two conclusions were reached as a result of this work. First, SREM in its current state can adequately specify software requirements for command and control systems. This was demonstrated during the CASD evaluation on critical command and control problems like the man-machine interface, database and communications. Also when SREM was lacking in a particular specification mechanism, it proved sufficiently flexible to enable its adaptation to new specification requirements, such as those exhibited by the man-machine command interface and database repeating group structures.

The evaluation did uncover some aspects of the SREM which were not completely responsive to command and control system needs. There are problems such as the need for examples of SREM applied to command and control systems. There are also oversights and omissions in the current terminology, training and documentation. These are not critical problems, and their resolution is straightforward.

Of a more serious nature are the problems encountered in SREM's ability to communicate stimulus-response oriented requirements to software designers,in its specification of software requirements for systems composed of several computers and in its use as an IVV methodology. There are also several aspects of SREM which are related to past "lessons learned" which were not addressed during the CASD evaluation. For example, the performance of the REVS system software, the effectiveness of the SREM training materials and the utility of the user interface to the SREM tools were not seriously addressed. All of these factors lead to the second conclusion. Further evaluation of SREM on large command and control systems is needed. This evaluation would definitize the problems uncovered in the in-house evaluation, expose any additional problem areas and recommend solutions.

SECTION 4

FURTHER SREM EVALUATION

The evaluation reported in the previous sections concluded that SREM was a viable approach for specifying command and control software requirements, but also recommended further in-depth investigation of SREM's capabilities in the context of specifying full-scale command and control systems. This investigation was contracted to Martin-Marietta's Denver Division in the Fall of 1980.

4.1 Objectives

The objectives of this work are to perform a technically thorough evaluation of the capabilities of SREM for specifying and analyzing command and control software requirements and to recommend improvements based on the evaluation results. The first phase of this work involved attending the SREM training sessions and evaluating the training course and its documentation. SREM was then used to validate the software requirements for a system which simulates command and control communication networks and also to develop the requirements for the sensor configuration assessment system, described in the previous section. Throughout these applications emphasis was placed on exposing

SREM deficiencies and building an experience base from which improvements could be proposed.

Martin-Marietta's evaluation will also include several other areas. One of them, communicating software requirements to software designers, will be approached by actually carrying out the structured design of one of the subsystems of the sensor configuration assessment system. It is hoped that sufficient experience can be gained during this design activity so that guidelines for improving this critical communications link can be developed. Another aspect to be investigated is how SREM should be integrated into the software acquisition process. Changes to the acquisition process as well as to SREM to adapt it to the process will be recommended. Specification products which can be used by software acquisition personnel at formal reviews will also be proposed. Finally, the evaluation will investigate the costs involved in using SREM on large software development projects, the performance of the SREM tools and the effectiveness of their user interfaces.

4.2 Interim Results

Interim results [DEMO81] of the Martin-Marietta evaluation of SREM have verified the in-house evaluation's conclusions - SREM is capable of specifying command and control software requirements, but certain improvements are needed. The interim results indicate that SREM and its specification language, RSL, can adequately represent:

decision processing, such as occurs between the analyst and database
real world objects, such as aircraft and vehicles
database elements

communications between subsystems and the data processor.

Aspects of SREM which have been identified as needing improvement include:

communications between parallel processes
hardware communications characteristics
real-time constraints involving hardware timing, interrupt handling and parallel processing
user training
communication of requirements to software designer or evaluator.

These interim results were based on the application of SREM to one command and control problem. They will be updated as required following the second application.

SECTION 6

SUMMARY AND CONCLUSIONS

The SREM has been shown to be a viable approach for specifying command and control software requirements Problems have been encountered, and the Rome Air Development Center has undertaken a program whose objectives are to fully investigate the problems and recommend their solutions. This will undoubtedly mean changes to the SREM, its specification language and analysis tools. The goals of this program are to utilize this promising technology to improve the quality of software requirement specifications and, by eliminating errors early in the software development process, reduce total system life cycle costs. In addition, such usage will enable assessment of the effectiveness of this kind of technology over a meaningful population of software development projects and will develop an experience base for the next step in the evolution of software development environments.

REFERENCES

[BELL76] Bell, T. E. and Thayer, T. A., "Software Requirements: Are They Really a Problem?" TRW-SS-76-04, TRW, Redondo Beach CA (July 1976).

[BOEH76] Boehm, B. W., "Software Engineering," TRW-SS-76-08, TRW, Redondo Beach CA (October 1976).

[CLAP77] Clapp, J. A. and Hazle M., "Building Blocks for C3 Systems," MTR-3504, MITRE Corp., Bedford MA (September 1977).

[DEMO81] DeMooy, B., Hartschuh, D. and Stone, A., "On The Applicability of The Software Requirements Engineering Methodology to Command, Control, Communications and Intelligence Systems: Interim Technical Report," Martin-Marietta Denver Aerospace, Denver CO (June 1981).

[GREW80] Grewe, J. C., Private communication. (May 1980).

[MCDO78] "Technical Report - SREM Evaluation Results Summary," MDC-G7750, McDonnell Douglas, Huntington Beach CA (October 1978).

[MILS68] "Specification Practices," Military Standard (MIL-STD) 490 (October 1968).

[SLEG] Slegel, R. C., "Applying SREM to the Verification and Validation of an Existing Software Requirements Specification," Johns Hopkins University Applied Physics Lab, Laurel MD (October 1977).

[STAI76] Stainer, H. M., "An Evaluation of PSL/PSA and RSL/REVS Two Computer-Assisted Software Requirements/Specification/Description Tools," FS-76-205, John Hopkins University Applied Physics Lab, Laurel MD (October 1976).

[USAF75] "Management of Computer Resources in Systems," Air Force Regulation 800-14, Vols. I and II, Department of the Air Force (September 1975).